# Recognition system of leaf images based on neuronal network

WANG Dai-lin, ZHANG Xiu-mei*, LIU Ya-qiu
*Northeast Forestry University, Harbin 150040, P.R. China*

**Abstract**: In forest variety registration, visual traits of the plants appearance are widely used to discern different tree species. The new recognition system of leaf image strategy which based on neural network established to administrate a hierarchical list of leaf images, some sorts of edge detection can be performed to identify the individual tokens of every image and the frame of the leaf can be got to differentiate the tree species. An approach based on back-propagation neuronal network is proposed and the programming language for the implementation is also given by using Java. The numerical simulations results have shown that the proposed leaf strategy is effective and feasible.

**Keyword**: Neuronal network; Edge detection; Leaf images; Pattern recognition
**CLC number:** TP39 **Document Code:** A **Article ID:** 1007-662X(2006)03-0243-04

## Introduction

The analysis of plant leaf growth for physiological coordinates has been of botanical interest for a long time (Maksymowych 1973; Spies *et al*. 2001). A botanical interpretation needs the data to be in a form that allows comparisons between different leaves. Therefore the results of the growth analysis have to be transformed into a coordinate system affixed to the leaf. Assessing the appearance of leaves is an important botanical skill. Any token of visual appearance is a potential character. These tokens on leaves can be extracted to specify tree species (Schurr *et al.*2000).

In forest variety registration, a recognition system can use leaf appearance to deal with many leaf species; at the same time this system should be easily extended to include an increasing set of varieties. Traditionally, tokens in leaf appearance can be assessed by measurement and subjective scoring. The former can be used for the measurements of simple size and shape, such as lengths and their ratios. While the latter is often used to scan more subtle differences which are more difficult to measure (Camlin *et al*. 1994).

The study on recognition system mainly focused on two aspects: one concentrates on seeking efficient methods to detect the leaf image and then identifying specific leaf image tokens, the other focuses on transferring the leaf image shape into a neuronal network. For this purpose, leaf recognition approach can be established by machine vision applied in the recognition field (Draper and Keefe 1988). These tokens will be the basis of the calculations of neuronal network and make it possible by recognizing an unknown leaf image to specify tree species.

In this paper, the leaves recognition approach is presented by using a neuronal network, which is based java application/applet to recognize images of leaves accordingly to previously trained the BP (back-propagation) network. A back-propagation network is a special form of a feed-forward neuronal network. It optimizes the learning process by propagating the weights of an actual neuron back to its previous matrix. An important feature of this approach is that it could be applied directly by a java enabled internet browser.

**Foundation project:** This paper was supported by National Natural Science Foundation of China (No. 30371126).
**Biography**: WANG Dai-lin (1974- ), female, master, assistant in Library of Northeast Forestry University, Harbin 150040, P. R. China.
**Received date**: 2006-03-23;
**Accepted date**: 2006-04-07;
**Responsible editor**: Song Funan
*Corresponding author: xiumeizhang1@126.com

## Mathematic model

### Image edge detection

One of the main tasks of this application is to detect the specific tokens in a leaf image. The edge detection is processed on the gray color amount of the pixels. Assuming that the image is a full 2D scan of a single leaf, the well-known prewitt edge detection algorithm is used in this application.

Prewitt edge detection produces an image where higher gray-level values indicate the presence of an edge between two objects. Prewitt edge detection filter computes the root mean square of two 3×3 templates.

Defining these two 3×3 templates to calculate the gradient value:

$$\begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix} \qquad \begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix} \tag{1}$$

X Y

Considering the following 3×3 image window:

$$\begin{vmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & a_9 \end{vmatrix}$$

where, $a_1$–$a_9$ are the gray levels of each pixel in the filter window.

$$X = -1*a_1 + 1*a_3 - 1*a_4 + 1*a_6 - 1*a_7 + 1*a_9 \tag{2}$$

$$Y = 1*a_1 + 1*a_2 + 1*a_3 - 1*a_7 - 1*a_8 - 1*a_9 \tag{3}$$

$$\beta = \sqrt{X^2 + Y^2} \tag{4}$$

where β is the Prewitt gradient. All pixels are filtered. In order to filter pixels located near the edge of an image, edge pixels values are replicated to give sufficient data.

**Thinning**

The outer frame of a leaf is enough to identify its species. Thus, it is necessary to identify this outer frame exactly. The previously applied prewitt edge detection normally just identify the edges with a pre-configured threshold and after that a thinning algorithm should be performed to minimize the threshold-based edge to a one-line frame where a sort of token recognition discussed later.

The thinning algorithm processed the image recursively and minimizes the found lines to a one-pixel width by comparing the actual pixel situation with specific patterns.

**Leaf image token**

The central part of this application is the tokens of each leaf image, which are found after the image processing.

The cosine and sinus angles of the shape represent the criteria of a recognition pattern in order to transfer the leaf image shape into a neuronal network usable form.

Fig. 1 shows a part of a leaf image that was already processed through the above-mentioned edge detection and thinning algorithms.

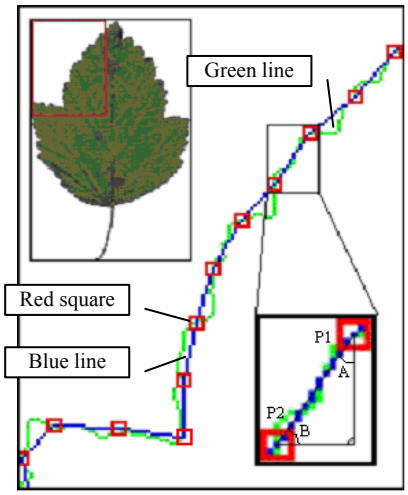


**Fig. 1 leaf image tokens**

**Note**: Green line: The shape of the leaf image after a successfully edge detection & thinning.

Red square: This square represents a point on the shape of the leaf image from which a line to the next square is drawn.

Blue line: The compound of the center of two squares from which the cosine and sinus angle are computed. Such a blue line is a representation of a leaf token.

The small right-angled triangle represents a token of a single leaf image. The summary of all triangles of a leaf image represents the tokens of a leaf from which the calculations of neuronal network can be started. Here it should be clear that the angles A and B are the two necessary parts which will be fit into the neuronal network layers. The direction of the hypotenuse can be exactly represented from point P1 to P2.

**The back propagation algorithm**

The most important part of this work is the integration of a feed-forward back-propagation neuronal network (Fig. 2) because it is a part of the task to show that just a back-propagation network and the shape of a leaf image are enough to specify the tree species. The implementation of network also just has one input, hidden and output layers to simplify and speed-up the calculation of java applet. The inputs for this neuronal network are the individual tokens of a leaf image, and as a token normally consists of a cosine and sinus angle, the amount of input layers for this network are the amount of tokens multiplied by two. The number of output neurons is normally specified by the amount of different species because we use an encoded form to specify the outputs. All other behaviors of the network are specified by the normal mathematical principals of a back-propagation network (Simon Haykin 2000).

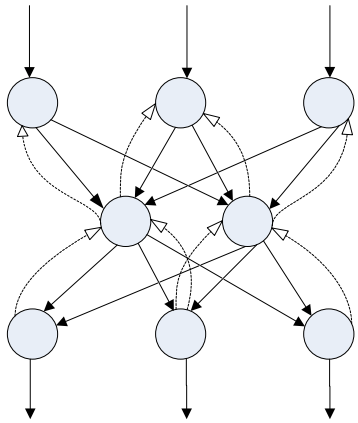

**Fig. 2 Feed-forward back-propagation neuronal network**

The network is studied by the mode of supervise. The back-propagation algorithm for multilayer networks is a generalization of the LMS algorithm, and both algorithms use the same performance index: mean square error. The algorithm is provided with a set of examples of proper network behavior:

$$\{(x_1, o_1), (x_2, o_2), \cdots \cdots (x_k, o_k)\} \tag{5}$$

The error equation can be defined as

$$E = \frac{1}{2}\sum_{k=1}^{l}[(d_k - O_k)^2] \tag{6}$$

where $x_k$ is the input to the network, $d_k$ the desired output, and $O_k$ the factual output. The $E$ is the sum of squares of all patterns. The algorithm should adjust the network parameters in order to minimize the mean square error. The gradient method is always used in minimization, and conjugate gradient method is used in some papers.

Steps of BP network's learning are as following:

1) to choose the pattern of training and suitable structure of network as well as preprocess the input data;

2) to set up all connection weights and thresholds of units with small random value;

3) to load all patterns to network orderly, and then give the desired outputs to network;

4) to calculate the output of hidden layers as formulation.

Because the error is an indirect function of the weights in the hidden layers, the chain rule to calculate the derivatives will be used. The first step is to propagate the input forward through the network. As discussed earlier, for multilayer networks the output of one layer becomes the input to the following layer. The net input to layer *j*, *k* are explicit functions of the weights and bias among input, hidden and output layer, respectively.

$net_j = \sum_{i=1}^{m} \omega_{ij} o_i + b_j$ is used between input and hidden layer,

$net_k = \sum_{j=1}^{q} \omega_{jk} o_j + b_k$ is used between hidden and output layer.

$o_j = f(net_j)$ and $O_k = f(net_k)$ are the output equation of hidden and output layer, respectively .

The sigmoid activation equation is $f(x) = \frac{1}{1+e^{-x}}$.

According to Delta learning principle,

$$\Delta \omega_{jk} = -\eta \frac{\partial E}{\partial \omega_{jk}} \tag{7}$$

Defining that $\delta_k = -\frac{\partial E}{\partial net_k}$,

Then Equation (3) can be calculated:

$$\Delta \omega_{jk} = -\eta \frac{\partial E}{\partial net_k} \cdot \frac{\partial net_k}{\partial \omega_{jk}} = -\eta \delta_k O_j \tag{8}$$

The same method was used to get

$$\Delta \omega_{ij} = -\eta \frac{\partial E}{\partial \omega_{ij}} = -\eta \frac{\partial E}{\partial net_j} \cdot \frac{\partial net_j}{\partial \omega_{ij}} = -\eta \delta_j O_i \tag{9}$$

Finally, to adjust the connection weights in turn,

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \alpha \Delta \omega_{ij}(t) \tag{10}$$

$$\omega_{jk}(t+1) = \omega_{jk}(t) + \alpha \Delta \omega_{jk}(t) \tag{11}$$

The network with random number is initialized to get the error rate from output neuron. In Eqs (.8) and (9),$\eta$ is a positive constant which calls learning rate. Learning rate is always less than 1. Much great learning rate can accelerate the rate of learning, but it can also lead to no guarantee converging. If learning rate is too little, number of iterations will increase obviously, and process of converging will slow down. In Eqs (10) and (11),$\alpha$is also a positive constant, which calls momentum factor. Increasing the valve of momentum factor can accelerate the process of converging and smooth the change of connection weights in the process of converging.

## Simulation results

To demonstrate the effectiveness of the proposed recognition system of leaf images, numerical simulations have been performed and presented in this section. For every class datum, about half of these have been correctly associated to primary training model and a quarter used to check up, the last quarter exists as the test. Perhaps the convergence can be sped by adjusting the learning rate during the course of training. The trick will be determined by changing the learning rate and by how much. The better result through the selected parameter will be obtained. After that, we can get the recognition information and result in Fig. 5. However, more training datum and extraction methods are needed to improve our results. The general process is illustrated as follows.

**Image processing**

One of the most important parts of the whole application is the image processing. (Cawkell.1994; Schmundt *et al.*1999).Without finding any useful tokens in the leaf images, the neuronal network calculation will stop. So we spent lots of our efforts in the edge detection and thinning algorithms.

The different operations were performed by a list of loaded images. The small left image in Fig. 3 is a view of the original image like in the image file and the big one in the center of this window is the image after pressing "Find Token" to process the different image processing operations including edge detection and thinning.

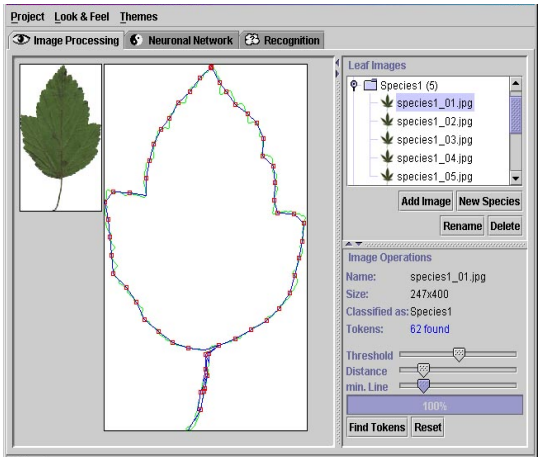


**Fig. 3 Image processing**

There are three configurable sliders on Fig. 3 can be used to define the threshold for the edge detection, the distance of the tokens (red square) and a minimum amount of pixels in a line should be recognized as a part of the shape.

The leaf images of the different tree species can be added and deleted from the hierarchical list at the right upper side of this window. So if adding an image to a species node, this image is recognized as part of this species and will be included in the neuronal network calculations. For a good quality result at least 5 images of a species should be added to give the neuronal network enough tokens to find the specific shape of this leaf species.

**Neuronal network**

Another main part of this application is the back-propagation neuronal network. The inputs for this neuronal network are the individual tokens of a leaf image, and as a token normally consists of a cosinus and sinus angle, the amount of input layers for this network are the amount of tokens multiplied by two.

Like in the first described "Image processing ", the "neuronal network tab" has also some sort of configuration. Mainly these configurations are the properties of the neuronal network.

For an instance, the following network information can be obtained from Fig. 3: there are 18 leaf images and 4 leaf species. Also there are 140 tokens got from the first step. We should keep the learning rate and number of hidden neurons low, because we normally get out a good training phase for the network. The network parameter is showed as Table 1. The amount of input neurons for the network is normally twice the amount of tokens because of the sinus & cosines value for one token. The amount of

output neurons is normally the amount of different leaf species that are in the hierarchical list of the image processing.

**Table 1. Network operations parameter**

| Input neurons | Hidden neurons | Output neurons | Learn rate | Momentum | Max. steps |
|---|---|---|---|---|---|
| 280 | 20 | 4 | 0.3 | 1.0 | 100 |

The result shows that when the step kept at 100, the error will reach about 0.0844. Based on the amount of images and network properties normally about 500–1000 training steps need to be trained to obtain a good result in the recognition later. If the error rate drops below 0.01, no problem will happen in recognizing different leaf images.

**Recognition**

The last of three parts in this application is the recognition figure where an image can be loaded to process recognition of the leaf image (Fig. 4). As the first "Image processing tab" people can control the threshold, distance and minline of the image processing that takes place on the loaded image as soon as pressing "Recognize".

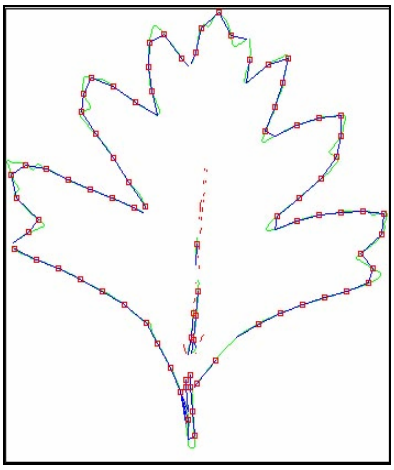

**Fig. 4 Recognition processing**

After this image processing, the leaves recognition application will use the recognized tokens of this new image as the values to identify the species. The results of this recognition will be displayed in a percentage, which shows the token of leaf is similar about 75.110%, 98.439%, 73.835%, and 49.871% to tree species 1 to 4. It is concluded that the tree species 2 in the trained network is most similar to the loaded leaf image.

## Conclusions

The new recognition system of leaf image strategy which based on back-propagation neural network was established to administrate a hierarchical list of leaf images by detecting some sorts of leaf tokens, which can be performed to identify the tree species in forest variety registration. A well-trained neuronal network normally can point at one specific species. Also the neighborhood of species can be explained with the results of the neuronal network of this application. This recognition system should test the robustness of the method on various images. In addition, iterations of the complete method with less strict parameters should be implemented, each step bringing new strong boundaries and new hypothesis reinforcements.

At the same time, the speed of implement stimulation on the computer should be improved. Finally, more training statistics will be needed to provide a conclusive confirmation to further pattern recognition. We intend to collaborate internationally with image analysis specialists and plant scientists to ensure the recognition is powerful, flexible and widely applicable.

## References

Camlin, M.S. and McMichael, A.C. 1994. New methodology for the measurement of leaf color in *ryegrass (Loliumspp)* [J]. Plant Varieties and Seeds, **7**: 37–49.

Cawkell, A. 1994. Managing image databases [J]. Image Processing, **6**: 36–39.

Draper, S. and Keefe, P.D. 1988. An automated machine vision system for the morphometry of new cultivars and plant gene bank accessions [J]. Plant Varieties and Seeds, **1**: 1–11.

Maksymowych. 1973. Analysis of leaf development [M]. Cambridge: Cambridge University Press

Schmundt, D. and Schurr, U. 1999. Plant leaf growth studied by image sequence analysis [C]. Computer Vision and Applications Volume 2 Signal Processing and Pattern Recognition. Academic Press, New York, Boston, London, Sydney, Tokyo, Toronto.

Schurr, U., Heckenberger, U., Herdel, K., Walter, A.and Feil, R. 2000. Leaf development in *Ricinus communis* during drought stress-dynamics of growth processes, of cellular structure and of sink source-transition [J]. Journal of Experimental Botany, **51**(350): 1515–1529.

Simon Haykin. 2000. Neural networks a comprehensive foundation [M], Second Edition. Beijing: China Machine Press, p112–123.

Spies, H., Scharr, H.and Schurr, U. 2001. Root growth analysis in physiological coordinates [C]. In: ICIAP'01, Palermo, Italy.